Sequence Planning for Robotic Assembly of Tetrahedral Truss Structures

Luiz S. Homem-de-Mello

Index terms: artificial intelligence
planning
assembly
tetrahedral truss structures
multihierarchical representation
hexagonal grid.

This research was conducted at the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration.

Luiz S. Homem-de-Mello is with the Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, 91109-8099.

Abstract

An artificial intelligence approach to planning the robotic assembly of large tetrahedral truss structures is presented. Based on the computational formalism known as *production system*, the approach exploits the simplicity and uniformity of the shapes of the parts and the regularity of their interconnection to drastically reduce the required geometric reasoning computation. The global database consists of a hexagonal grid representation of the truss structure. This representation captures the multiple hierarchies in tetrahedral truss structures and allows a substantial reduction of the search space without sacrificing completeness. It allows the choice of a hierarchy to be made only when needed, thus allowing a more informed decision. Testing the preconditions of the production rules is computationally inexpensive because the patterned way in which the struts are interconnected is incorporated into the topology of the hexagonal grid representation. A directed graph representation of assembly sequences allows the use of both graph search and backtracking control strategies. The extension of the approach to planning repair sequences is outlined. A prototype planner, named TASP, has been implemented and successfully generated assembly sequences for a structure made of 102 struts.

1 Introduction

Fig. 1 shows a tetrahedral truss structure similar to those that will be used in future space missions [1 4]. The assembly, disassembly or repair of these large tress structures requires careful planning in order to guarantee that the parts are assembled, or disassembled, in a correct and efficient sequence. This planning is needed regardless of whether the assembly is executed by humans or by robots.

Because of the size and complexity of these truss structures, even trained humans may fail to detect *dead-end* sequences until a lot of work has been done and it is found that the overall assembly cannot be completed. In the case of a repair in which a faulty strut is to be replaced, an ill-planned disassembly sequence may lead to an irreparable collapse of the whole truss structure.

In addition to the difficulty humans have in guaranteeing correctness in the planning process, they often fail to notice which possibilities for the sequences are the most efficient. This difficulty is further aggravated by constant changes in the measure of the efficiency of the assembly sequence. For example, the efficiency may be measured by the total time it takes to complete the assembly in one case, and by the total energy in another case.

Moreover, humans typically are slow in generating assembly sequences. There are many situations in which the sequence planning must also be expeditious. Speed in sequence generation is particularly important in the case of a repair in which a faulty strut is to be replaced. It is virtually impossible to preplan for every conceivable repair that may be needed. Speed in sequence generation is also important in the design of the truss structures because it allows the difficulty of assembly to be considered in the design process. A designer may also want to take into account the difficulty of repairing different structures.

This paper presents an artificial intelligence approach to planning the assembly of large tetrahedral truss structures based on the computational formalism known as *production system*. The approach exploits the simplicity and uniformity of the shapes of the parts and the regularity of their interconnection to drastically reduce the required geometric reasoning computation. In addition, the approach uses a multihierarchical representation that allows a substantial reduction of the search space without sacrificing completeness.

The robotic assembly facility of the NASA Langley Research Center [14, 17] has been used as the reference scenario. Fig. 2 shows that facility in schematic form. The robot arm is mounted on a base which is mounted on a carriage that can translate along one

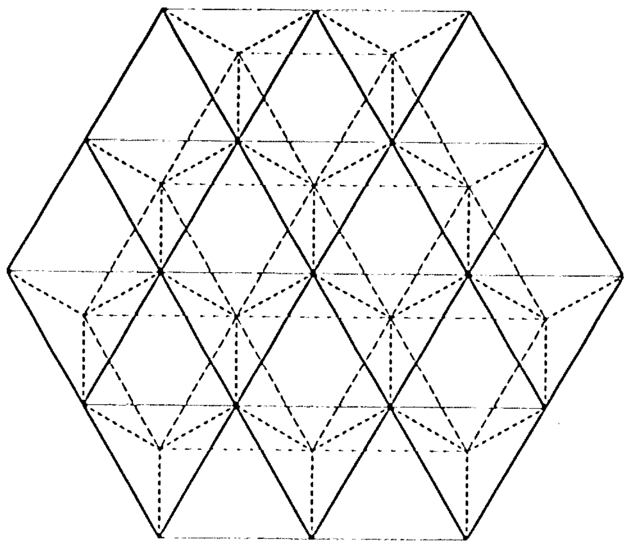

Figure 1: Tetrahedral truss structure.

direction. The base where the robot is mounted can translate along a direction orthogonal to the carriage translations. 'J']) (xc two motions allow the positioning of the robot arm in a Cartesian coordinate: system. The truss structure is mounted on a base that can rotate. If necessary, before a strut is assembled, the structure is turned and both the base of the robot arm and the carriage are translated.

The assembly process consists of a succession of tasks, each of which is the addition of one strut. The process starts with all struts stored in pallets that are stacked on the same base where the robot arm is mounted. The assembly process ends with all s(,ı uts properly joined to form the whole structure. Ideally, aft er struts have been added, they are not removed until the end of the assembly process.

An assembly task is said to be feasible if there is a collision-free path to bring the stint, to its position in the structure from a situation in which it is far apart, and if it is possible to lock the joints that attach the strut to its nodes. Of course, the path should also avoid collisions between the robot arm or the carriage and the t russ structure.

2 1 Background

Most, previous work on assembly sequence planning [1, 2, 4, 5, 6, i', 8, 9, 15, 19, 20] focused on electromechanical and electronic devices such as gearboxes, alternators, and disk drives.

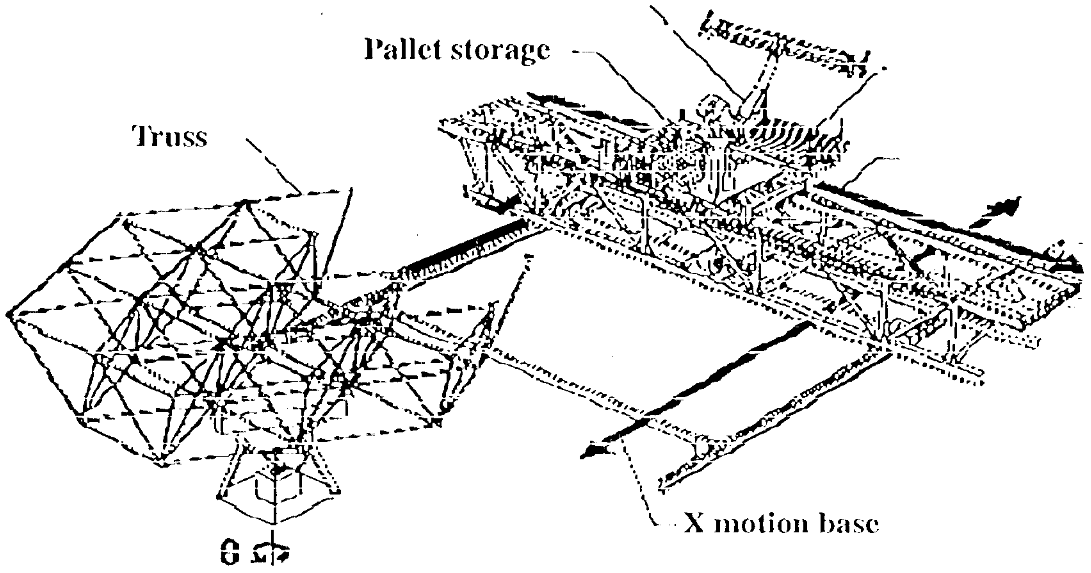

Figure 2: Robotic assembly facility at the NASA Langley Research Center.

The difficulty in planning the assembly sequence for those products stems, in some degree, from the variety of part shapes and from the lack of regularity in the way the pieces are interconnected. To overcome this difficulty, previous approaches used elaborate representations of mechanical assemblies and complex geometric and symbolic reasoning techniques.

Another difficulty in the automation of assembly sequence planning for electromechanical and electronic devices crones from the fast growth in the required computation with the increase in the number of parts. Previous approaches have overcome this problem by clustering components into subassemblies [9], thereby artificially reducing the number of parts. Many large products have natural subassemblies that arise as a result of modular design as well as of manufacturing advantages. Clustering components into subassemblies sacrifices completeness since sequences that interleave the assembly of parts of different subassemblies cannot be generated. But for most large products this 10ss of completeness is not a serious limitation because those natural subassemblies are assembled independently anyway. In practice, a hierarchical model of the assembly is used to implement the clustering of parts. At the highest level, each subassembly is treated as a part.

Case-based reasoning has been used recently [11, 12] and successfully generated plans by using a database of previously solved cases to solve a similar new problem.

Despite the recent progress in assembly planning, it is still impracticable to use misting

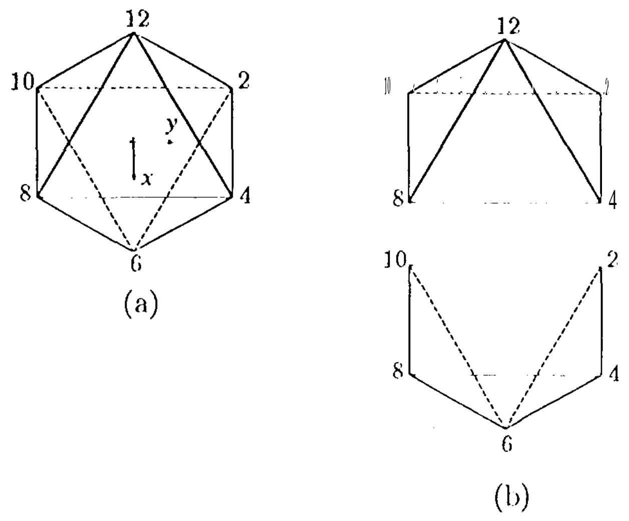

Figure 3: A small truss structure and its subdivision into two pentahedral units.

planners to generate an assembly sequence for assemblies containing a large number of parts, such as the structure shown in Fig. 1, which is made of 102 struts, because of the complexity of the reasoning involved and the large size of the solution space.

One way to reduce the computation, when planning the assembly of tetrahedral truss structures, is to cluster the struts into subassemblies. A truss structure such as the one shown in Fig. 1 can be viewed as the composition of tetrahedral and pentahedral units, much like a solid that has a complex shape can be treated as the composition of simple solids that have faces in contact, one against the other. Two adjacent units share the struts and nodes of their "contacting" faces. For example, the small truss structure shown in Fig. 3a can be regarded as the composition of the two pentahedral units shown in Fig. 3b. Those two pentahedrons have one face "in contact" and they share the struts and nodes of that face. Similarly, the structure shown in Fig. 1 can be viewed as a composition of tetrahedral and pentahedral units with faces "in contact." Since in practice it is preferred to finish the assembly of one unit before beginning the assembly of another [13], viewing the structure as a composition of units should not be a problem in the assembly sequence planning.

One problem with using a hierarchical approach for planning the assembly of tetrahedral truss structures is that there are several ways to cluster the struts into those kinds of unit. For example, in addition to the subdivision shown in Fig. 3b, there are two other ways in which the small truss structure shown in Fig. 3a can be subdivided into two pentahedral

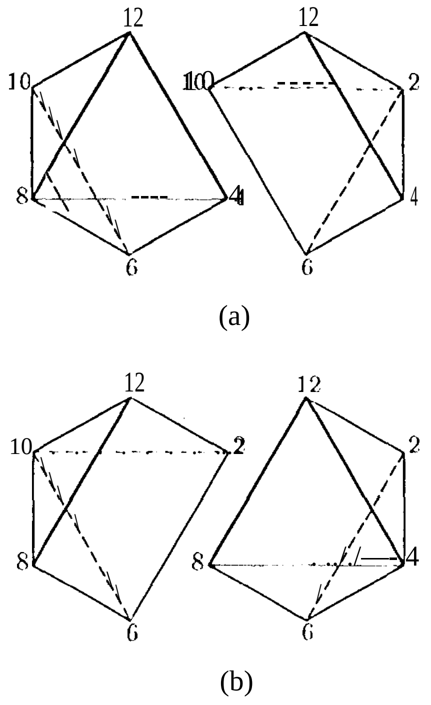

Figure 4: Two additional subdivisions of the small tress structure (Fig. 3a) into two pentahedral units.

units and these are shown in Fig. 4. Unlike electromechanical and electronic devices, in the case of truss structures there is no manufacturing advantage in choosing one subdivision over the others. Instead of having one natural hierarchy of the parts, tetrahedral truss structures have several hierarchical models none of which is "more natural" than the others. When the small structure shown in Fig. 3a is part of a large structure (e.g., Fig. 1), unless the best assembly sequence is known in advance, choosing one of its subdivisions to create a hierarchical model will likely preclude the generation of the best assembly sequence. Therefore, it is important that the representation of the problem captures the many distinct hierarchies that occur in a tetrahedral truss structure.

A second way to reduce the computation, when planning the assembly of tetrahedral truss structures, is to take advantage of the simplicity and uniformity of the shapes of the parts and the regularity of their interconnection. Unlike electromechanical and electronic devices, the tetrahedral truss structures are made of struts, all of which have the same cylindrical shape. Moreover, those struts are interconnected in a regular fashion. Because the parts have the same shape and are interconnected in a patterned way, the model of a truss structure can explicitly incorporate the geometry of the set of parts.

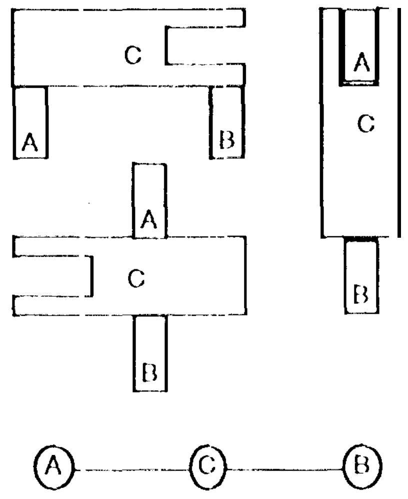

Figure 5: Three assemblies with different geometry but same topology.

The models of assemblies that have been used in previous work describe the shapes of all parts and the geometric and mechanical relationships between parts. Typically, assembly models can be associated with graphs in which the vertices correspond to the parts and the edges to the geometric relationships between parts. The topology of the graph corresponds to the topology of the parts in the assembly. But there is no relation between the geometry of the set of parts in the assembly and the topology of the graph. Fig. 5 shows three simple assemblies made up of the same set of parts. Those assemblies are associated with the same graph, also shown in Fig. 5, since the topology of the parts is the same. But the geometry of the parts in each assembly is very different from the geometry of the parts in the others.

3 Planning the assembly of tetrahedral truss structures

The computational formalism known as production system [](] has been used for the automatic generation of assembly sequences for tetrahedral tress structures. There are three major elements in a production system: the global database, the set of production rules, and the control scheme. This section describes these three elements, outlines the extension of the approach to plan repair sequences, and dicusses the main aspects of a prototype that has been implemented.

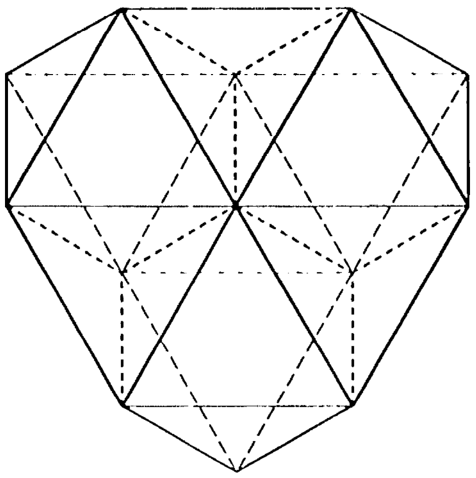

Figure 6: A subset of the structure shown in Fig. 1.

3.1 A multihierarchical representation of tetrahedral truss structure

The representation to be introduced next is based on viewing tetrahedrons and octahedrons as the building blocks of a tetrahedral truss structure. A pentahedron will be considered to be a building block only when it is not embedded in any octahedron. As it will become clear below (after Fig. 9), pentahedrons not contained in any octahedron occur at the periphery of a structure. Fig. 6 shows a subset of the structure depicted in Fig. 1, and Fig. 7 shows its building blocks.

As discussed above, there are six embedded pentahedrons in an octahedron. The nodes of the octahedron shown in Fig. 3a have been numbered by analogy with the numbers in a clock face. The embedded pentahedrons are refered to by the letter "P" followed by the number of the vertex corresponding to their aim. Therefore, the pentahedrons shown in Fig. 3b are referred to as P12 (top) and P6 (bottom); the pentahedrons shown in Fig. 4a are referred to as P8 (left) and P2 (right); and the pentahedrons shown in Fig. 4b are referred to as P10 (left) and P4 (right).

Fig. 3a shows a coordinate frame associated with an octahedron. The xy plane contains nodes 2, 6, and 10. The z axis points out of the figure, that is, nodes 4, 8, and 12 have positive z coordinate. The octahedrons in a truss structure are all parallel to each other. Therefore, the transformation between the coordinate frames of two octahedrons is a pure translation.

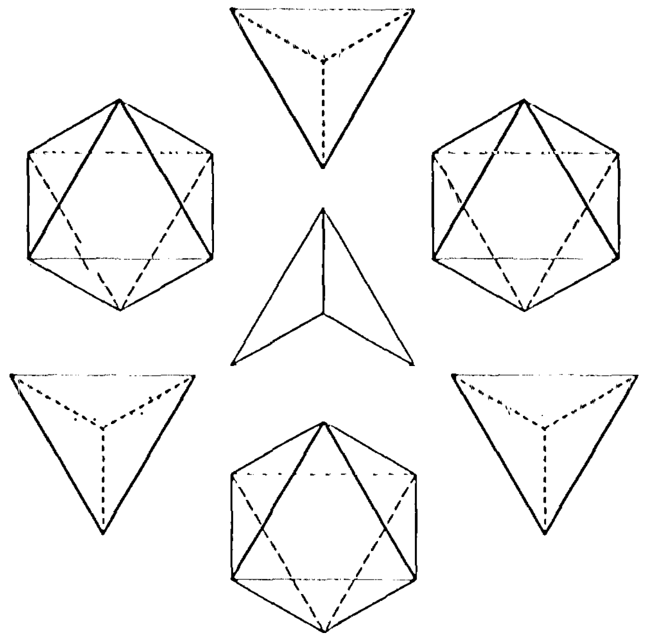

Figure 7: The building blocks of the structure shown in Fig. 6.

Some tetrahedrons have three nodes on the top plane and one node on the bottom plane. These are referred to as *tetrahedron-down* because they can be viewed as a pyramid pointing down. The other tetrahedrons, which have three nodes on the bottom plane and one node on the top plane, are referred to as *tetrahedron-up*. Fig. 8 shows the coordinate frames associated with each type of tetrahedron. The z axis points out of the figure. The xy plane contains nodes 1, 2, and 3 in the case of tetrahedron-up (Fig. 8b), and only node 4 in the case of tetrahedron-down (Fig. 8a). All tetrahedrons-up in a truss structure are parallel to each other, and all tetrahedrons-down are parallel to each other. Therefore, the transformation between the coordinate frames of two tetrahedrons-up (or two tetrahedrons-down) is a pure translation.

A tetrahedral truss structure can be represented by a graph in which the vertices correspond to volumetric units, and the edges correspond to "face-contacts" between adjacent units. Fig. 9 shows a graph representation for the 102-strut truss structure shown in Fig. 1.

The geometry of this graph parallels that of the truss structure. Because of the regularity of the structures, their graph representation constitute a hexagonal grid. In addition, the hexagonal grid can be mapped into a rectangular grid as shown in Fig. 9, where the lines and columns are labeled with their indices. Furthermore, a coordinate frame can be associated

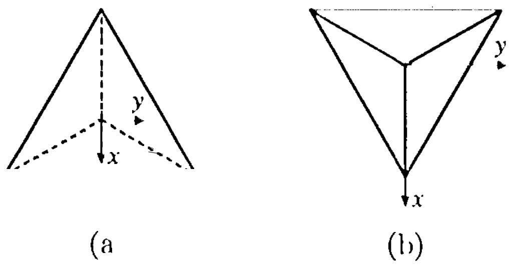

Figure 8: The coordinate frames associated with each type of tetrahedron: (a) tetrahedron-down; (b) tetrahedron-up.

with the graph shown in Fig. 9. The graph's x axis, which is parallel to the x axis of all units, points down. And the graph's y axis, which is parallel to the the y axis of all units, points right.

There are three types of vertices, represented, respectively, by hexagons, triangles, and half-hexagons. Hexagon vertices correspond to octahedrons like the one shown in Fig. 3a. Triangles correspond to tetrahedrons; triangles pointing down in the figure correspond to *tetrahedrons-down*, and triangles pointing up correspond to *tetrahedrons-up*.

Unlike regular graphs, in this representation the position and orientation of the vertices are importan . A coordinate frame is associated with 1e graph. Its axes are parallel to those of the frames associated with the octahedrons. Each vertex is oriented as ts unit's parallel projection on the bottom plane of the structure.

The half-hexagons cor espond to pentahedrons such as hose shown in Figs. 3b, 4a, and 4b. The half-hexagons are used only when there is no octahedron that includes the cor esponding pentahedron. Since there are six pentahed ons embedded in each octahedron, there are six orientations in which he half-hexagons may occur, and hey are shown in Fig. 10.

The edges in he graph representation of a truss structure correspond to those "faces" that are common to two adjacent volumetric units, hat is, those sets of three struts that "belong" to bot volumetric units.

The mapping of the graph representation into a ectangular grid gives rise to a data-structure for a computer implementation: a two dimensional array in which each element may contain information about one building block of the truss structure. The indices of the array element indicate the position of the building block.

The edges in the graph shown in Fig. 9 are only implicitly encoded into the two-

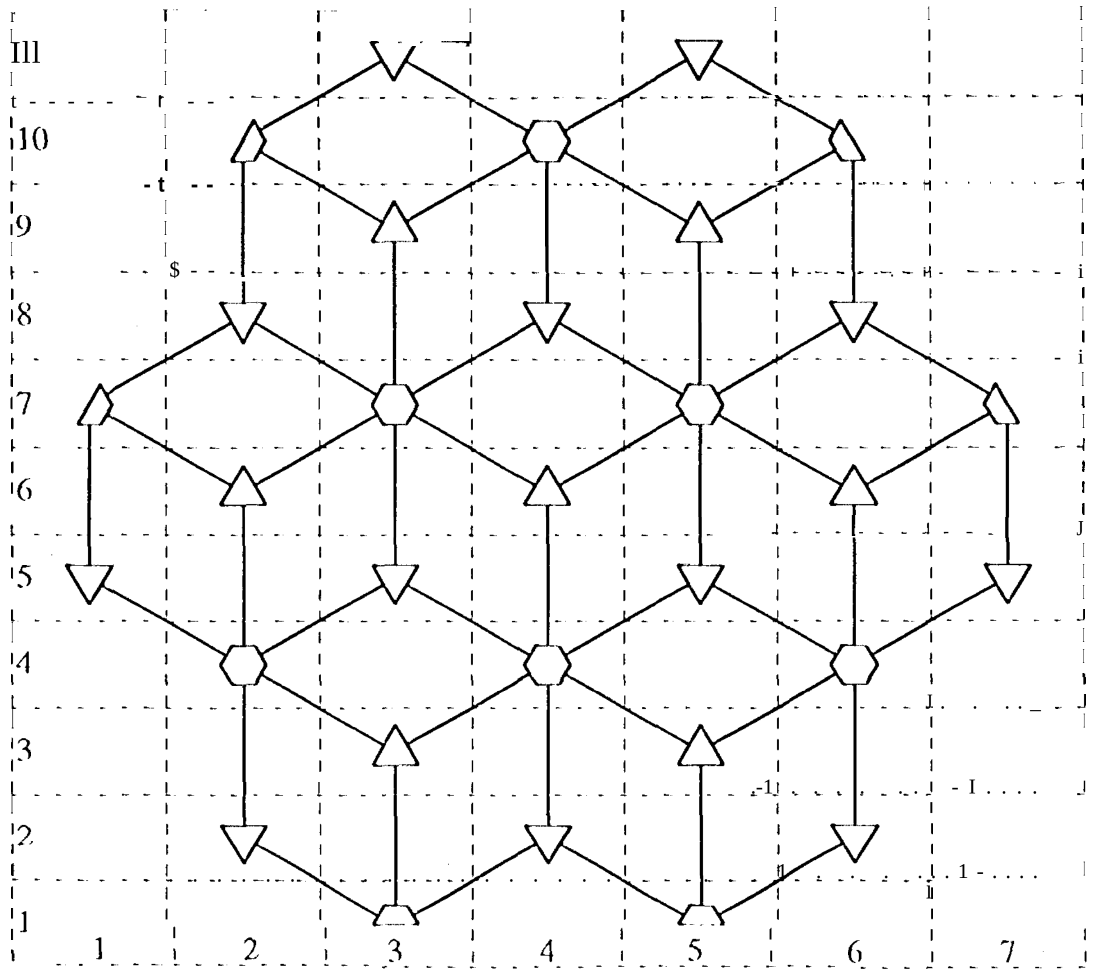

Figure 9: Graph representation for a tetrahedral truss structure with 102 struts and its mapping into a rectangular grid.

dimensional array data structure. In addition, the contacts between units that share only one strut, or only one node, are also implicitly encoded into the array. For example, a tetrahedron-up at cell (i, j) (i.e. line i, column j) shares one strut with the tetrahedron-down at cell $(i+2, j)$, another strut with the tetrahedron-down at cell $(i-1, j-1)$, and another strut with the tetrahedron-down at cell $(i-1, j+1)$. As another example, an octahedron at cell (i, j) shares one node with the tetrahedron at cell $(i+2, j-2)$.

It should be pointed out that Fig. 9 shows one mapping from the graph representation of tetrahedral truss structure, which is a hexagonal grid, into a quadratic grid. The quadratic grid shown in Fig. 9 has the advantage of being rectangular, but it leaves many empty calls. In a computer implementation, if the available storage space is scarce, it is straightforward

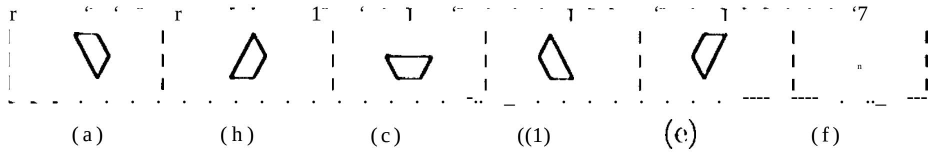

Figure 10: The six orientations in which half-hexagons may occur: (a) P2; (b) P4; (c) P6; (d) P8; (e) P10; (f) P12.

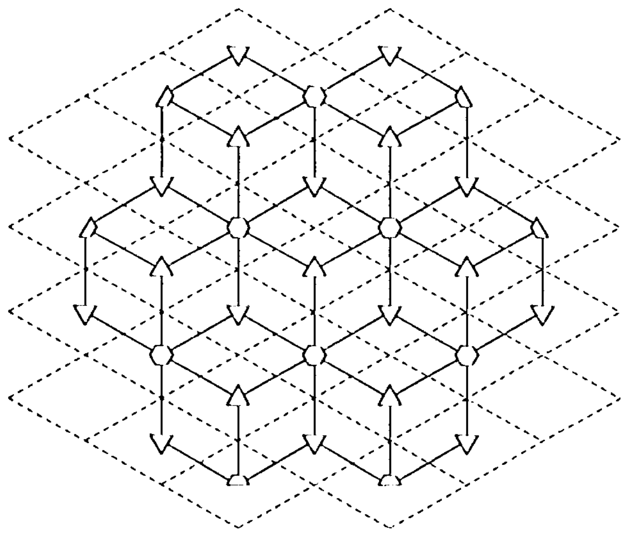

Figure 11: The mapping of the graph representation of tetrahedral truss structures into a quadratic (not rectangular) grid.

to devise other mappings from the hexagonal grid into a quadratic grid, which may not be rectangular. Fig. 11 shows another mapping from the graph representation of truss structures into a quadratic grid. Unlike the one shown in Fig. 9, the mapping shown in Fig. 11 does not leave empty cells, except, of course, in the periphery of the graph.

As it will become clear in the following subsections, this graph representation of truss structures allows an assembly planner to exploit the regularity in which the parts are joined to improve its planning efficiency. This improvement is due, in part, to the encoding of the geometry of the truss structure in the topology of the graph. Moreover, the graph representation also allows the planner to take advantage of the multiple hierarchies that exist in tetrahedral truss structures. The decision of which hierarchy to choose does not have to be made until it is needed. Being able to delay the selection of the hierarchy, the planner will have more information available to decide which hierarchy is more advantageous, and

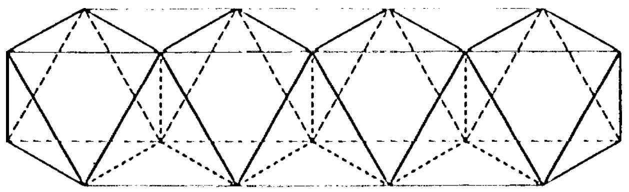

Figure 12: A linear tetrahedral truss structure.

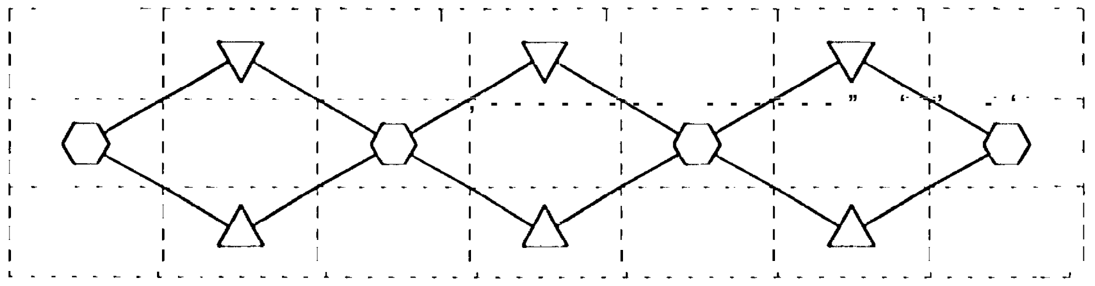

Figure 13: The graph representation of the linear tetrahedral beam shown in Fig. 12.

therefore will be able to make a better choice.

As an example of an another configuration, Fig. 12 shows a linear tetrahedral truss structure. The graph representation of this structure is shown in Fig. 13.

3.2 Control strategy

Several methodologies for representing assembly sequences have been utilized [8], including representations based on directed graphs and AND/OR graphs.

As mentioned in Section 3, it is preferred to complete the assembly of a tetrahedral or pentahedral unit before beginning the assembly of another unit [13]. Therefore, the assembly task can be redefined as the assembly of one tetrahedron or one pentahedron. In this definition, each assembly task consists of a sequence of subtasks, each being the assembly of one strut.

Since each assembly task is the addition of exactly one volumetric unit, both the directed graph and the AND/OR graph will have the same size. The directed graph representation has been used in this work because it is simpler and easier to understand and implement.

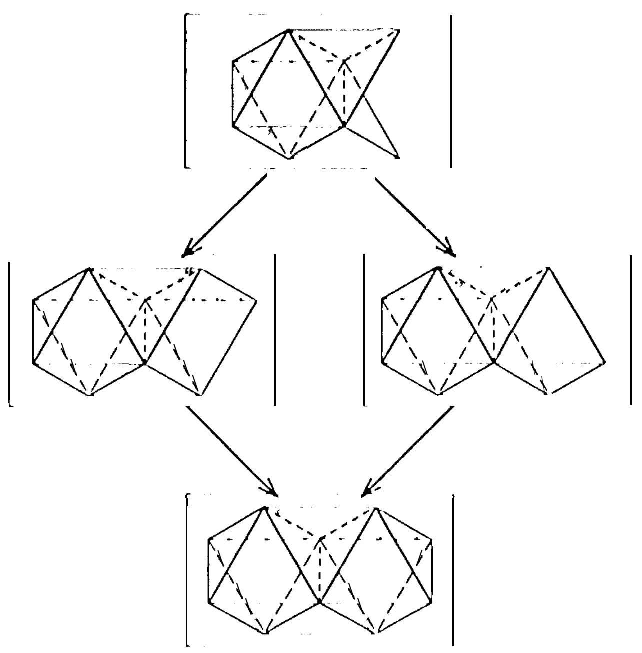

Figure 14: A portion of the directed graph of assembly sequences, which is also shown in Fig. 15. The vertices have been labeled by the top view of the partial truss structure at each State of the assembly process.

The vertices in this directed graph correspond to the states of the assembly process, which can be characterized by the description of the substructure already assembled. The edges in this directed graph represent the assembly tasks, each corresponding to the addition of one volumetric unit.

Figs. 14 and 15 show a portion of the directed graph of assembly sequences. in Fig. 14, the vertices have been labeled by the top view of the partial truss structure at, each state of the assembly process. This labeling is better for displaying the assembly sequences for humans. In Fig. 15 the vertices have been labeled by the graph representation of the partial truss structure at each state of the assembly process. This labeling reflects more closely the computer internal representation of the assembly sequences. In both figures, the vertex at the top corresponds to a state in which one octahedron and two tetrahedrons are already assembled. The two vertices in the middle corresponds to states in which an additional pentahedron is already assembled. In the left, vertex the additional pentahedron is P10, and in the right vertex the additional pentahedron is P8. The vertex at the bottom corresponds to a state in which two octahedrons and two tetrahedrons are already assembled.

Figs. 14 and 15 also illustrate the advantage of using the multihierarchical representation

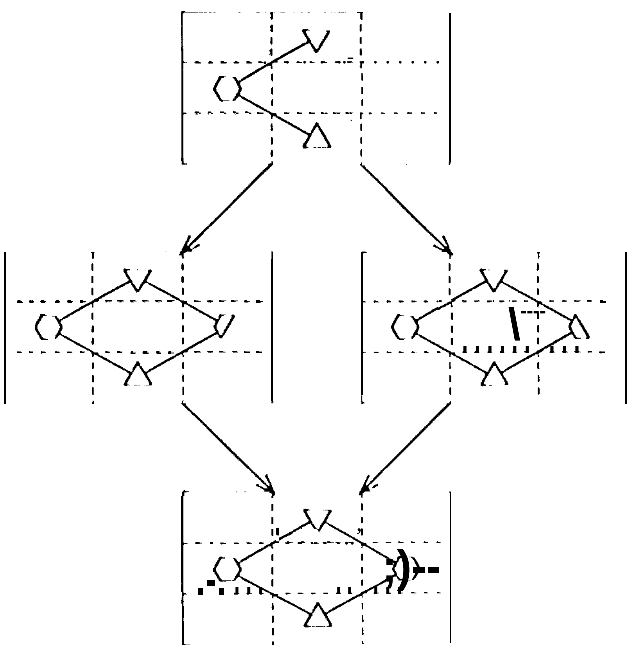

Figure 15: A portion of the directed graph of assembly sequences, which is also shown in Fig. 14. The vertices have been labeled by the graph representation of the partial truss structure at each state of the assembly process.

of tetrahedral truss structures introduced in Section 3.1. Because the building blocks are tetrahedrons and octahedrons, it is possible to generate sequences that use different sets of pentahedrons as assembly tasks. As pointed out above, the additional pentahedron in the left middle vertex is not the same as the one in the right middle vertex. By using the representation on Fig. 9, the three possibilities in which an octahedron can be subdivided can be considered. In the scenario described in Section 1, the two possibilities corresponding to the subdivision of the right octahedron into P6 and P12 are not considered valid. If the structure had been viewed as a composition of pentahedrons and tetrahedrons, only one alternative would be considered.

Each assembly sequence corresponds to a path in the directed graph of assembly sequences, starting in the vertex that has no label (i.e., no strut has been assembled) and ending in the vertex that is labeled by the whole truss structure (e.g., the structure shown in Fig. 1). By construction, the directed graph of assembly sequences has no cycle. A measure that reflects the quality of an assembly sequence can be computed by assigning costs to the vertices (i.e., the states of the assembly process) and to the edges (i.e., the assembly tasks).

The cost of a path p can be defined recursively as:

$$cost(p) = \begin{cases} C_S(s_p) & \text{if the path has only one node} \\ C_S(s_p) + C_T(t_p) + cost(r_p) & \text{otherwise} \end{cases}$$

where s_p is the initial vertex (state) of p, t_p is the initial edge (task) of p, and r_p is the tail of p, that is, what is left of p after s_p and t_p are removed. The function C_S gives an assessment of the quality of a state of the assembly process. Better (e.g., more stable) states correspond to smaller values of C_S. The function C_T gives an assessment of the quality of a task in the assembly process. Better (e. g., less complex or less time consuming) tasks correspond to smaller values of C_T.

An additional advantage of the directed graph representation of assembly sequences and its associated cost function is that it allows both backtracking and graph search control regimes [10] to be implemented. In the backtracking control regime, only one path is maintained in the computer memory. This scheme uses less memory, but it may become inefficient if the same path is explored more than once. In the graph search control regime, all the partial paths explored are kept in memory. This scheme requires more memory, but avoids repeating the same computation when a path is explored more than once.

The construction of the assembly sequence can proceed in backward or forward fashion. The former is easier to understand because it corresponds to the actual assembly process. But the assembly planning is usually more efficient going backwards, from the goal state to the initial state, because it avoids dead-md states.

3.3 Production rules

The production rules that are introduced in this subsection contain the conditions for the execution of an assembly task and the changes that occur in the state of the tress structure when that task is executed. In the operation of the planning system, whenever a production rule is applied, the global database must be updated to reflect the changes in the state of the truss structure.

A production rule has two parts: precondition and effect. The precondition specifies the situations in which the rule can be applied. The effect describes the changes that occur in the global database when the rule is applied.

The simplest way to introduce the production rules is by an example. Fig. 16 shows one production rule. It corresponds to the assembly task that finishes up one octahedron, starting with one of its pentrahedral halves already assembled. This production rule is

- Precondition:
  1. Cell (i,j) currently contains a pentahedron Pk.
  2. Goal is one octahedron in cell (i,j).
  3. All cells (x,y) for which $L(x,y,i,j,k) > 0$, where

     $L(x, y, i, j, k) = -\cos\alpha \cdot x + \sin\alpha \cdot y + i \cdot \cos\alpha - j \cdot \sin\alpha$,

     are empty
- Effect:
  1. Adjust the angle of the truss structure and the xy position of the robot arm according to the position of cell (i,j).
  2. Install pentahedron Pk' in cell (i,j), where $k' = \text{rem}(6 + k, 12)$.

Figure 16: Production rule example; $\alpha = \arctan\frac{b(k)}{a(k)}$, where a($k$) and b($k$) are defined in Table 1. Fig. 17 shows a state that satisfies the precondition of this production rule.

associated with the case in which only the base of the "missing" pentahedron is in place. Therefore, the assembly task will include the assembly of the four struts from the base to the apex node.

If the pentahedron already assembled is Pk, the pentahedron that will complete the octahedron is Pk', where $k' = \text{rem}(6+k, 12)$. Since k can be any of the six elements of {2, 4,6,8,10,12}, Fig. 16 can be viewed as describing six production rules.

The first two preconditions simply verify that the goal is an octahedron in a cell (i, j) that currently has a pentahedron

The third precondition verifies that no collision will occur between the truss structure and the carriage where the base of the robot is mounted. It requires that all cells on the same side as Pk' with respect to the line $L(x,y,i,j,k) = 0$ be empty. This line has an orientation, like an axis, and goes through cell (i,j). The angle from the horizontal axis of the graph representation of the truss structure to line $L(x,y,i,j,k) = 0$ is $\alpha = \arctan\frac{b(k)}{a(k)}$. This is the angle by which the horizontal axis must be turned to become aligned and have the same orientation as the line $L(x,y,i,j,k) = 0$. Table 1 shows the pairs a(k) and b(k) for each value of k.

The third precondition subsumes the precondition that the tetrahedrons in the cells

Table 1: The pairs a(k) and b(k) such that $\frac{b(k)}{a(k)}$ is the tangent of the angle from the horizontal axis of the graph representation of the truss structure (Fig. 9) to line $L(x, y, i, j, k) = 0$ in the production rule example shown in figure 16.

k	a(k)	b(k)
2	1	-3
4	-1	-3
6	-1	0
8	-1	3
10	1	3
12	1	0

(5, 5), (6, 6), and (8, 6) have not yet been assembled. This guarantees that only the base of the "missing" pentahedron is in place.

Fig. 17 shows a state that satisfies the precondition of the production rule shown in Fig. 16 for cell (7, 5) and $k = 10$. The line $L(x, y, 7, 5, 10) = 0$ is also shown.

The effect of this production rule is the installation of pentahedron Pk' in cell (i, j). Before this can be executed it may be necessary to turn the truss structure and to move the carriage and the base of the robot arm. This adjustment depends on the position of cell (i, j). The actual installation of struts follows a precompiled sequence of subtasks each of which is the addition of one strut. This precompiled sequence, which includes the motions of the robot arm, is independent of the position of the cell (i, j).

For cell (7, 5) and $k = 10$ in Fig. 17, the effect of the production rule shown in Fig. 16 is the addition of a pentahedron P4 since $k' =$ [illegible]((i + k, 12) = 4. Therefore, the assembly task includes the assembly of the four struts incident to node 4.

The production rule shown in Fig. 16 is used when the generation of assembly sequences goes in forward fashion. It is straight forward to write a corresponding production rule for generating assembly sequences in backward fashion, or disassembly sequences.

For each possible geometric configuration that a cell can take, there is a production rule

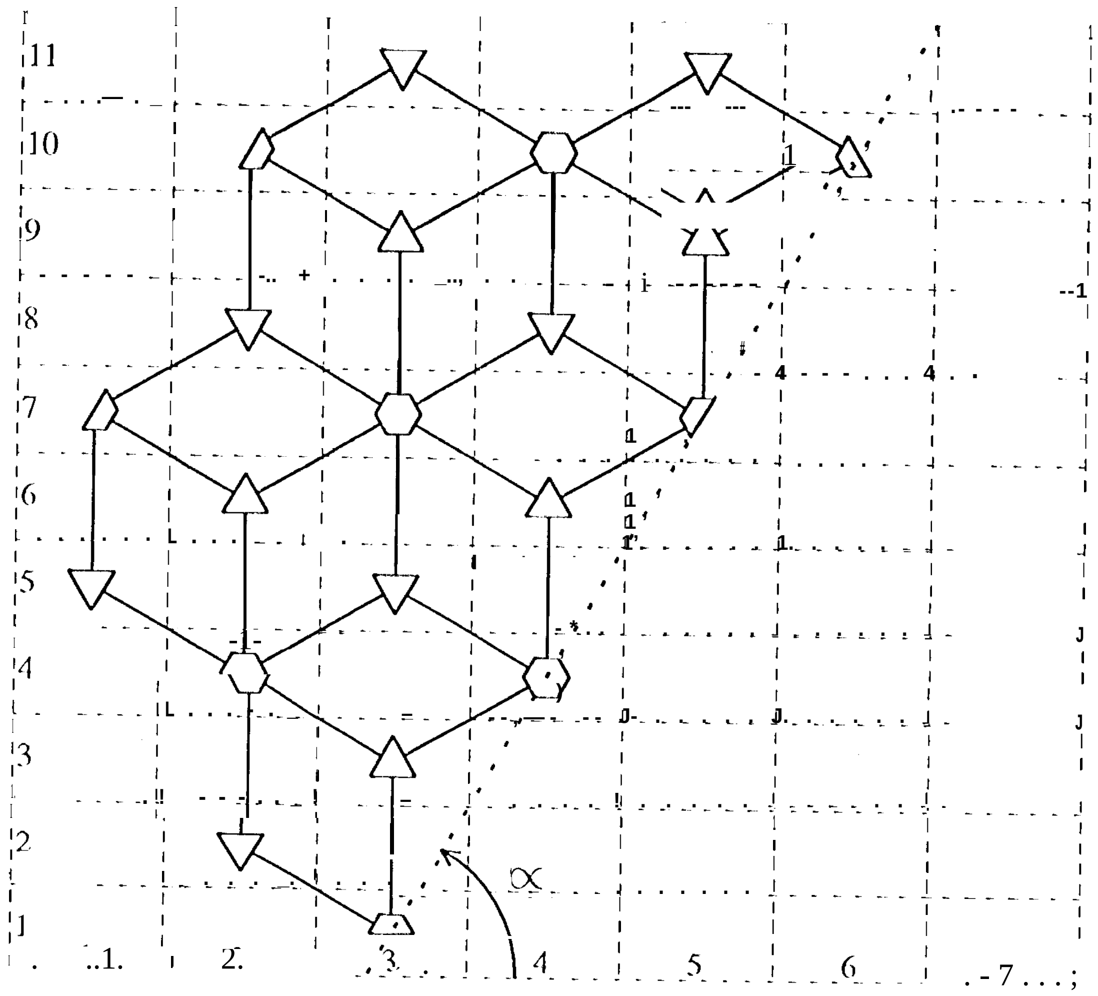

Figure 17: Example of state that satisfies the precondition of the production rule shown in Fig. 16 for cell (7, 5), and $k = 10$.

similar to the one in Fig. 16. Since there are only a few geometric configurations for a cell, the total number of production rules is small.

3.4 Planning repair sequences

The approach described in this section is adequate for planning the assembly of a truss structure. It may also be used to plan the disassembly of the whole structure. In either case, the planing strategy can take advantage of the fact that the number of tasks in all the alternative assembly sequences is the same, since there must be one task for each strut in the truss structure.

In the case of repair, however, there is no need to remove all struts. When planning the replacement Of a strut, the approach described in the previous subsections must be modified to also minimize the number of struts removed, that is, to minimize the number of tasks. This can be accomplished by using the production rules in a different way. Instead of using the production rules to test whether or not the addition of a unit is feasible, they are used to find the largest substructure to which a unit that contains the target, strut may be added.

Fig. 17 shows the largest structure to which a pentahedron 1'4 can be added to cell (7, 5). This substructure can be determined by finding out which cells must be empty if the precondition of the production rule shown in Fig 16 is to be satisfied.

The replacement of one of the four struts from the base to the apex of pentahedro 1'4 in cell (7, 5) can be accomplished by first removing the struts in the units not shown in Fig. 17. Once those struts are removed, it is possible to disassemble the pentahedro 1'4 in cell (7, 5). At this point the new stint, is replaced and the whole structure can be assembled by following the reverse of the disassembly sequence.

Since the same strut can be part of up to six units, the planning of repair sequences must consider up to six largest substructures. Furthermore, all the production rules that can be used to complete each type of unit must be considered. In the case of repair, the planning system must consider all alternatives and select the overall best.

3.5 Prototype system

In order to test the effectiveness of the approach and representation described above, an interactive production system named TASP has been implemented. To provide insight into good cost functions, TASP is interactive, uses a backtracking control scheme, and operates in forward fashion[1].

The input to TASP is a description of the desired structure. In addition) the first unit to be assembled must also 1)(! given. At each step, a menu containing all the subunits that can be assembled next is displayed for the user. These options are obtained by testing whether or not the preconditions of the production rules match the current state of the assembly process. The alternatives in the menu are ranked according to TASP's preference criterion[2]. The user may accept the system's choice for the next subunit or may select

[1]As mentioned in subsection 3.2, both backtracking and graph search control regimes can be implemented. Furthermore, the construction of the assembly sequence can proceed in backward or forward fashion. The choice of the most appropriate control regime will depend on the particular application, as will the selection of the direction in which the sequence is generated.

[2]The cost function that is used is a function of the translation of the carriage, the translation of the base and the rotation of the structure. The shorter those motions, the lower the cost function. The task for which

another alternative among those that are feasible. A graphical display of the truss structure allows the user to visualize the available options. At any point, the user can force TASP to backtrack and to "undo" one or more assembly tasks.

Although primarily a research tool, TASP has been used to generate assembly sequences for the 102-strut structure shown in Fig. 1. Assembly sequences generated by TASP significantly reduced the amount of rotation when compared to a sequence generated by hand. As an engineering tool, this interactive system exploits the strengths of humans and computers. Computers are better at guaranteeing that the sequence is correct and that no option is overlooked. Humans are better at assessing the quality of an assembly sequence. Furthermore, the use of this interactive system has provided insight into the key aspects that a cost function must capture.

In each assembly task, a number of struts are assembled. For example, in the tasks corresponding to the effect of the production rule shown in Fig. 16, four struts are assembled. By properly positioning the carriage and the base of the robot, the arm motions to install a given strut are the same regardless of the position of the octahedron that is being completed. In the current implementation, these motions were taught [3]. Each production rule is associated with the paths to install the struts of its corresponding subunit. Therefore, the output of the planning system includes, for each strut, the positions of the carriage and the base of the 1'01.)()(, the angle of the truss structure, and the specific arm motions to be used.

4 Conclusion

In previous work [15], the representation used for assemblies consisted of a hierarchy of three levels. The highest (i.e., the most abstract) level described the topology of the parts. Whenever an assembly is made of parts that have the same shape and that are connected in a patterned way, the highest level of its representation can incorporate not only the topology of the parts but also their geometry. In the case of a truss structure, the representation can also capture its multiple hierarchies.

By using a representation that incorporates the geometry of the parts in its highest level, the planner can avoid the complex geometric reasoning that is necessary to decide whether or not a candidate assembly task is feasible. Instead of performing computation comparable to that of path planning algorithms in order to decide the feasibility of a candidate assembly task, the planner tests the preconditions of a set of production rules. These tests can be

the cost function is minimal has the highest preference.

accomplished with inexpensive computation. The production rules correspond to the possible geometric configurations. Typically a small number of rules cover all possibilities. For tetrahedral truss structures there are only 12 rules[3]. The same production rules, employed in a different way, can be the basis for planning repair sequences which require the partial disassembly.

By using a hierarchical representation, the size of the search space can be significantly reduced. In the case of truss structures, there are different ways to cluster the parts into subassemblies but none is naturally preferable than the others. The representation of tetrahedral truss structures introduced in this paper captures their multiple hierarchies. The loss of completeness that usually arises with the use of hierarchical representations is not a problem in this case. By using a multihierarchical representation, the choice between hierarchies can be made as the plan is generated, thus allowing a better selection than if the choice were made in advance.

To deal with devices in which only a subassembly is made of identical parts interconnected in a regular way, this approach can be combined with the other existing approaches to assembly planning.

The use of a prototype system, which is interactive, is providing further insight into the production rules and into the key aspects that a cost function must capture. Future work will focus on cost functions and their corresponding heuristic evaluations.

Future work should also explore the combination of the approach described in this paper with machine learning techniques. For applications in which the cost function is fixed, and the configuration of the structure is always the same (e.g., it is always hexagonal), there may be direct enumeration schemes which would avoid the need for search techniques.

References

[1] D. F. Baldwin, T. E. Abell, M. M. Lui, T. L. De Fazio, and D. E. Whitney. *An Integrated Computer Aid for Generating and Evaluating Assembly Sequences for Mechanical Products. IEEE Transactions on Robotics and Automation*, vol. 7, No. 1, pp. 78 94, February 1991.

[2] A. Bourjault. *Contribution a une Approche Méthodologique de L'Assemblage Automatisé: Elaboration Automatique des Séquences Opératoires.* Thèse d'état, Université de Franche-Comté, Besançon, France, November 1984.

[3] J. J. Craig. *Introduction to Robotics.* Addison-Wesley, 1986.

[3] As pointed out in subsection 3.3, because of symmetry, each production rule actually covers either 3 or 6 possibilities. The 12 rules cover 54 possibilities, still a small number

[4] T. L. De Fazio and D. E. Whitney. Simplified Generation of All Mechanical Assembly Sequences. *IEEE Journal of Robotics and Automation*, RA-3(6):640–658, December 1987. Corrections *ibid* RA-4(6):705–708, December 1988.

[5] J. M. Henrioud. *Contribution a la Conceptualisation de l'Assemblage Automatisé: Nouvelle Approche en vue de Détermination des Processus d'Assemblage.* Thèse d'état, Université de Franche-Comté, Besançon, France, December 1989.

[6] J. M. Henrioud and A. Bourjault. *Détermination des Arbres d'Assemblage.* R. A. I. R. O. APII, vol. 24, pp. 547–564, November 1990.

[7] L. S. Homem de Mello. *Task Sequence Planning for Robotic Assembly.* PhD thesis, Carnegie Mellon University, May 1989.

[8] L. S. Homem de Mello and A. C. Sanderson. Representations of Mechanical Assembly Sequences. *IEEE Transactions on Robotics and Automation*, 7(2):211–227, April 1991.

[9] S. Lee and Y. G. Shin. Assembly Planning Based on Subassembly Extraction. In *Proc. IEEE Int. Conf. on Robotics and Automation.* IEEE Computer Society Press, May 1990.

[10] N. J. Nilsson. *Principles of Artificial Intelligence.* Tioga, 1980.

[11] P. Pu. An Assembly Sequence Generation Algorithm using Case-based Search Techniques. In *Proc. 1992 IEEE Int. Conf. on Robotics and Automation*, pp. 2425–2430. IEEE Computer Society Press, May 1992.

[12] P. Pu, and M. Reschberger. Assembly Sequence Planning Using Case-Based Reasoning Techniques. *Knowledge-Based Systems*, 4(3):123–130, September, 1991.

[13] M. D. Rhodes. *Guidelines for Development of Truss Assembly Scenario.* Unpublished technical notes, March 1990.

[14] M. D. Rhodes, R. W. Will, and M. A. Wise. *A Telerobotic System for Automated Assembly of Large Space Structures.* NASA Technical Memorandum 101518, Langley Research Center, Hampton, VA, March 1989.

[15] A. C. Sanderson, L. S. Homem de Mello, and H. Zhang. Assembly Sequence Planning. *AI Magazine*, 11(1):62–81, Spring 1989.

[16] R. W. Will et al. Hardware Testbed Experience in Automated Assembly of Space Structures. In A. Desrochers, Editor, *Intelligent Robotic Systems for Space Exploration*, Kluwer Academic Publishers, 1991.

[17] R. W. Will and M. D. Rhodes. An Automated Assembly System for Large Space Structures. *Cooperative Intelligent Robotics in Space*, Rui J. deFigueiredo, William E. Stoney, Editors, Proc. SPIE 1387, 60–71, 1991.

[18] R. W. Will and M. D. Rhodes. Preliminary Test Results and Upgrades for an Automated Assembly System. *Cooperative Intelligent Robotics in Space II*, William E. Stoney, Editor, Proc. SPIE 1612, 1991.

[19] R. Wilson. *On Geometric Assembly Planning* PhD thesis, Stanford University, March 1992.

[20] R. Wilson, and J. Latombe. *Reasoning About Mechanical Assembly.* Report No. STAN-CS-92-1442, Department of Computer Science, Stanford University, September 1992.